



# PAT Tutorial

## **Current and future PAT Task Force contributors**

**Roberto Tenchini, Frederic Ronga, Giovanni Petrucciani, Steven Lowette, Luca Lista, Petar Maksimovic, Sal Rappoccio, Eric Vaandering, Wolfgang Adam, Slava Krutelyov, Volker Adler, Christian Autermann, Jeremy Andrea, Clemens Zeidler, Colin Bernet, Benedikt Mura, Christophe Delaere, Tanja Rommerskirchen, Gheorghe Lungu, Freya Blekman, Benedikt Hegner, Jean-Roch Vlimant, Petra Van Mulders, Gregory Hammad, you?,...**



# Fine Print

Physics  
Analysis  
Toolkit

## .Current and future status

- ◆ We are attempting to migrate to 2.0.x and 2.1.x
- ◆ Obviously this incurs some cost
- ◆ I'm going to tell you about the "state of the art" in the talk, but the tutorial is not
- ◆ Much of this was already covered in the Run Plan Workshop, but today we can go into more detail and have more one-on-one time at the end
- ◆ Based on 1.6.12
  - Small enough to actually fit on the CRAB :)



# What's in it for you?

Physics  
Analysis  
Toolkit

## .Why do you have to use the PAT?

- ♦ This isn't quite asking the right question.
- ♦ The right question is "What can the PAT provide for me?"
- ♦ Answer:
  - Common vetted tools
  - Less coding time
  - Easier path to blessing
- ♦ Are you restricted to what is in the PAT?
  - Heavens, no!
  - Please feel free to develop use cases that the PAT does not consider
  - Also feel free to get them put into the PAT
- ♦ These are tools for all to use. We're here to make your life easier.

.This is capitalism, not autocracy. If you have a better product,  
we want to incorporate it.



# The Physics Analysis Toolkit



## •Outline

- ◆ the Candidate model
- ◆ the Physics Analysis Toolkit
- ◆ the Starter Kit
- ◆ conclusions



## .What is a Candidate?

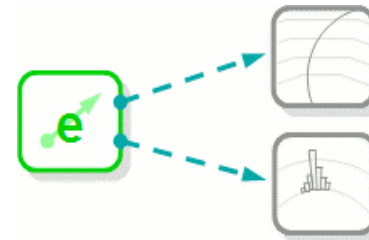
- ◆ Represented by a vertex and a four-vector
- ◆ Has particle ID information
- ◆ Anything that has a four-vector and a vertex inherits from Candidate



- Jets
- Electrons
- Photons
- Muons
- Missing Et



- ◆ Candidates can have detector objects as data members



- ◆ Support for lists of candidates



- ◆ Can also have composite candidates





# Overview of the Candidate Model

## Navigation

- ◆ Candidates have “mothers” and “daughters”
- ◆ For instance, Monte Carlo particles are Candidates (GenParticle), have access to full linked list of ancestry and progeny

```
const GenParticle & higgs = getHiggsFromSomewhere();  
for( size_t i = 0; i < higgs.numberOfDaughters(); ++ i ) {  
    const Candidate * Z = higgs.daughter( i );  
}
```

- ◆ Composite candidates “act” exactly the same way!
  - In above example, replace GenParticle with CompositeCandidate, and you’re done!

```
const CompositeCandidate & higgs = getReconstructedHiggs();  
for( size_t i = 0; i < higgs.numberOfDaughters(); ++ i ) {  
    const Candidate * Z = higgs.daughter( i );  
}
```

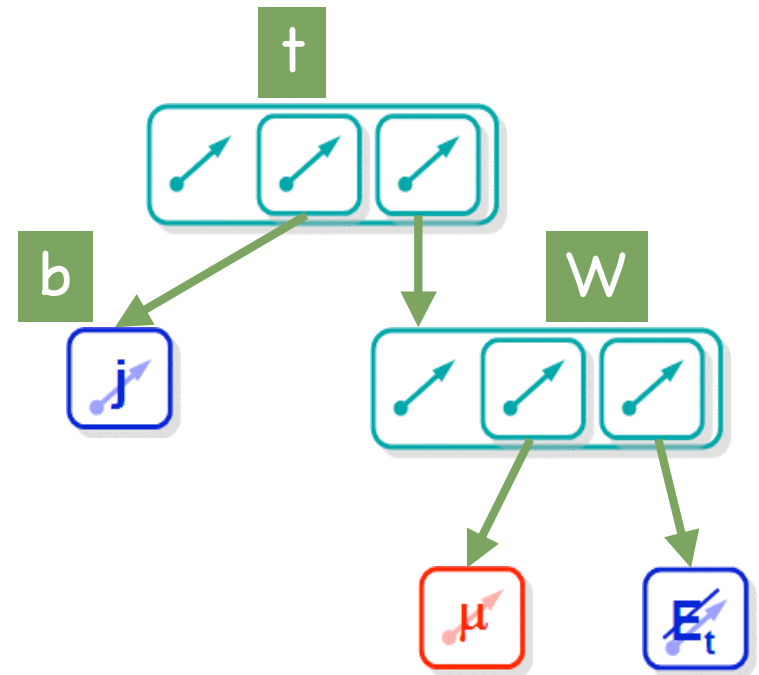


# Overview of the Candidate Model

Physics  
Analysis  
Toolkit

## .Hierarchical structure

- ◆ Access hypotheses in an intuitive way
- ◆ Easy interface for I/O of your objects
- ◆ Can have shallow clones for storing kinematics of fitted objects
- ◆ Utilize common plotting utilities to save you time



.Ultimately a time saver!



# Overview of the Candidate Model

## .Changes to candidate types

- ◆ Previously Candidate landscape was a little cluttered
- ◆ We changed that! ( $\geq 2.0.9$ ,  $\geq 2.1.0\text{pre6}$ )
- ◆ The only things you need to worry about are
  - “by value”: `Candidate` (base Candidates)
  - “by pointer”: `Ptr<Candidate>` (smart pointers to Candidates)
  - “by shallow clone pointer”: `ShallowClonePtrCandidate`  
(`Ptr<Candidate>` which has different four-vector and vertex, suitable for fitting, etc)
- ◆ Composite Candidates have also subsumed functionality of “`NamedCompositeCandidates`”
  - Can now add names and roles to `CompositeCandidate`, `NamedCompositeCandidate` is deprecated



# Overview of the Candidate Model

Physics  
Analysis  
Toolkit

## .Candidate Selectors

- ♦ Many selectors overall
  - Pt, eta, delta R, pdg id, etc
- ♦ Mostly limited to things available to candidate
- ♦ Can use the same functionality to make more advanced selectors
  - This is a little cumbersome to do

## .Where's the code?

- ♦ Some examples:
  - <http://cmslxr.fnal.gov/lxr/source/PhysicsTools/CandSelectors/plugins/>
  - <http://cmslxr.fnal.gov/lxr/source/PhysicsTools/RecoAlgos/plugins/>
- ♦ The real drivers:
  - <http://cmslxr.fnal.gov/lxr/source/PhysicsTools/UtilAlgos/interface/SingleObjectSelector.h>
  - <http://cmslxr.fnal.gov/lxr/source/PhysicsTools/UtilAlgos/interface/ObjectSelector.h>

```
module goodMuons = CandSelector {  
    InputTag src = allMuons  
    string cut = "pt > 5.0"  
}
```



# Overview of the Candidate Model

## .Candidate Combiners

- ◆ CandViewCombiner
  - Combines by "value"
    - Outputs View<CompositeCandidate> with value clones of leaves
- ◆ CandViewShallowClonePtrCombiner
  - Combines by "shallow clone pointer"
    - Outputs View<CompositeCandidate> with shallow clones of leaves
- ◆ Okay, so the names are yucky. But these are the only two you need.

## .Where's the code?

- ◆ PhysicsTools/CandAlgos/interface/CandCombiner.h
  - plugin definition
- ◆ PhysicsTools/CandUtils/interface/CandCombinerBase.h
  - Workhorse for the actual combinations
- ◆ PhysicsTools/CandUtils/interface/CandCombiner.h
  - Driver for workhorse

```
module zToMuMu = CandViewShallowCloneCombiner {  
  string decay = "muons@+ muons@-"  
  string cut = "50 < mass < 120"  
  string name = "zToMuMu"  
  vstring roles = { 'mu1', 'mu2' }  
}
```



# The Physics Analysis Toolkit



## •Outline

- ◆ the Candidate model
- ◆ the Physics Analysis Toolkit
- ◆ the Starter Kit
- ◆ conclusions



# Overview of the PAT

Physics  
Analysis  
Toolkit

## •PAT: what is it about?

- ♦ bridge to close the gap between the rough AOD and physics plots
  - we start from the needs from the physics user's perspective
- ♦ common set of basic tools for almost any physics analysis in CMS
  - no code duplication and faster development with a common tool
  - easier, faster and shared problem solving and debugging
  - code reliability enhanced → easier approvals of the results
  - easy to switch from one to another analysis
- ♦ provide sensible defaults and configurations
  - always have a consistent workflow through the PAT with very few includes
  - allows to quickly bootstrap an analysis chain
- ♦ full FWLite compatibility
- ♦ support two almost contradictory requirements
  - maximize the flexibility of the tools
  - maximize the user-friendliness of the tools





# Overview of the PAT

Physics  
Analysis  
Toolkit

## •PAT: what it is not about

- ◆ we don't re-invent the wheel
  - we don't rewrite the AOD objects
    - All objects inherit from Candidate!
    - Have full access to Candidate utilities discussed
  - POG specific tasks remain with POGs, we provide an interface to POG's algorithms (providing feedback!)
  - PAG specific algorithms remain within the PAG codebase (but we can provide infrastructure)
- ◆ the PAT is not a framework on it's own
  - fully embedded in the EDM: no private rootuples are written
  - profit from framework's persistency and provenance tracking
- ◆ we don't claim to cover all analysis use cases
  - but if you need/have extra tools, let us know and contribute!



# Overview of the PAT

## • The PAT in layers

### ♦ event interpretation

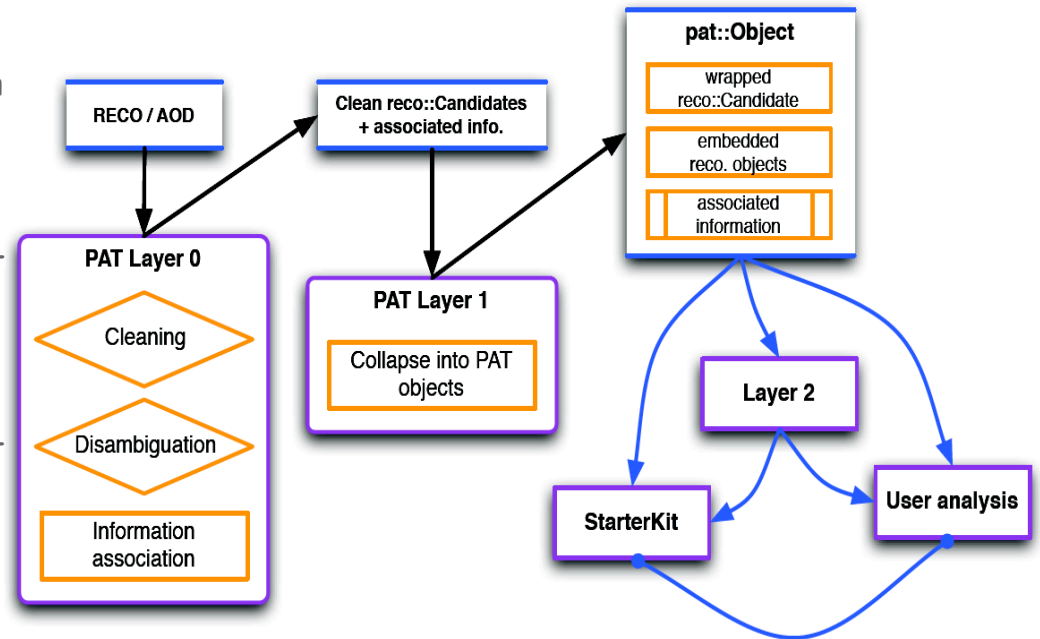
- “Layer 0”: cleaning and disambiguation  
→ event interpretation + additional analysis-level tasks (e.g. MC+trigger matching)
- “Layer 1”: creation of big objects that collapse externally associated information → no algorithmic tasks

### ♦ event hypothesis

- “Layer 2”: event hypothesis dependent tasks → provide possibility for re-tuning event interpretation

### ♦ analysis

- “Starter Kit”: for data exploration and plotting
- “paste-your-analysis-here”





# Overview of the PAT

Physics  
Analysis  
Toolkit

## .Output of PAT

**The input of the PAT is AOD/RECO.  
The output of the PAT is your business.**

- ♦ We give tools for many output options:
  - Skimming
  - Slimming
  - Four vectors of three variables (or all)
  - Text files
  - Your PRL (in progress\*\*\*)
- ♦ We do not constrain any of them. It is entirely between you, your PAG/POG, and the other members of your analysis

**.This is not intended to be another large scale layer of production**

\*\*\* I'm kidding, in case that wasn't clear. :)



# The Physics Analysis Toolkit



## •Outline

- ◆ the Candidate model
- ◆ the Physics Analysis Toolkit
- ◆ the Starter Kit
- ◆ conclusions



# The Starter Kit

- **Starter Kit Overhaul, 2.0.x and 2.1.x**
  - ♦ Changed directory structure
    - Moved plugins .cc from "test" to "plugins"
    - Moved plugins .h from "test" to "interface"
  - ♦ Added PatAnalyzerSkeleton (Freya Bleckman)
    - FKA "VerySimplePatAnalysis"
  - ♦ Renamed "StarterKit" plugin to "PatAnalyzerKit"
  - ♦ Extended usage of CompositeKit
  - ♦ Moved specific kits that depend on other software to appropriate packages
    - TtSemiEvtKit -> TopQuarkAnalysis
    - Zgamma\*Kit -> ?????
  - ♦ StarterKit no longer depends on anything aside from PAT and Candidate tools, to remove physical couplings



# The Starter Kit

Physics  
Analysis  
Toolkit

## .PatAnalyzerSkeleton

- ◆ FKA “VerySimplePatAnalysis” plugin from Freya Bleckman
- ◆ Shows how to pull out PAT objects and loop over them simply

## .PatAnalyzerKit

- ◆ FKA “StarterKit” plugin
- ◆ Plots everything in the event and has a simple ntuple interface

## .CompositeKit

- ◆ Plots an input collection of composite candidates



# The Starter Kit

Physics  
Analysis  
Toolkit

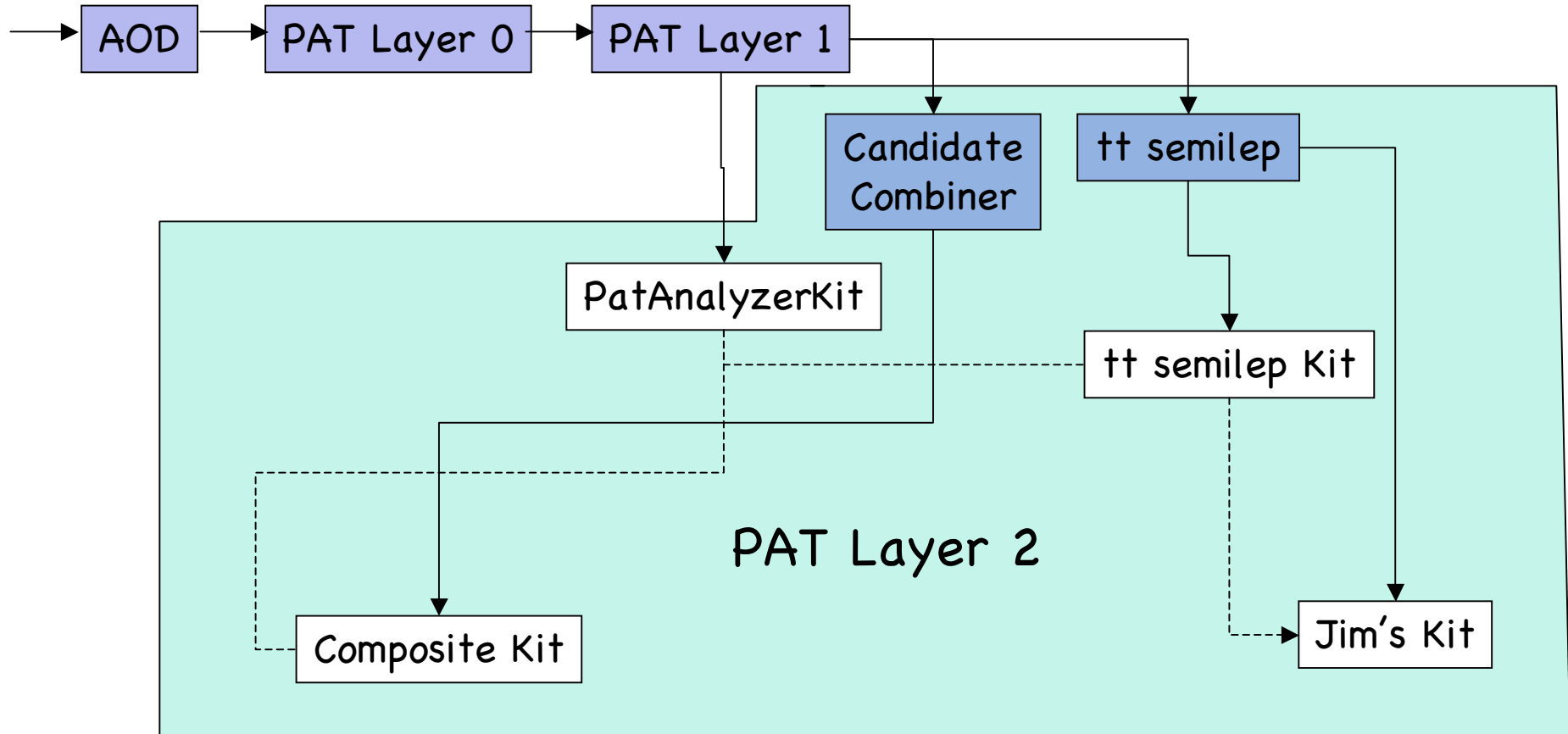
## • Analysis Starter Kit Goals

- ♦ a plotting package for standard plots of standard objects
  - allows quick exploration of data without lots of coding and bookkeeping
  - goal is to provide several analysis examples, monitored for physics changes
  - make histograms and simple rootuples out of PAT Layer-1 or Layer-2 objects
  - user can easily add plots (requiring some coding, obviously)
- ♦ the Starter Kit is ideal to make you step into CMS analysis and get to do some physics quickly

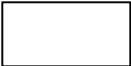


# Starter Kit Overview

Physics  
Analysis  
Toolkit



 = "calculating" kit

 = "plotting" kit

———— = Data flow

----- = Inheritance





# The Starter Kit

## • The Starter Kit in practice

- ♦ examples in the CMS Workbook

- <https://twiki.cern.ch/twiki/bin/view/CMS/WorkBookAnalysisStarterKit>

- ♦ what it does for you

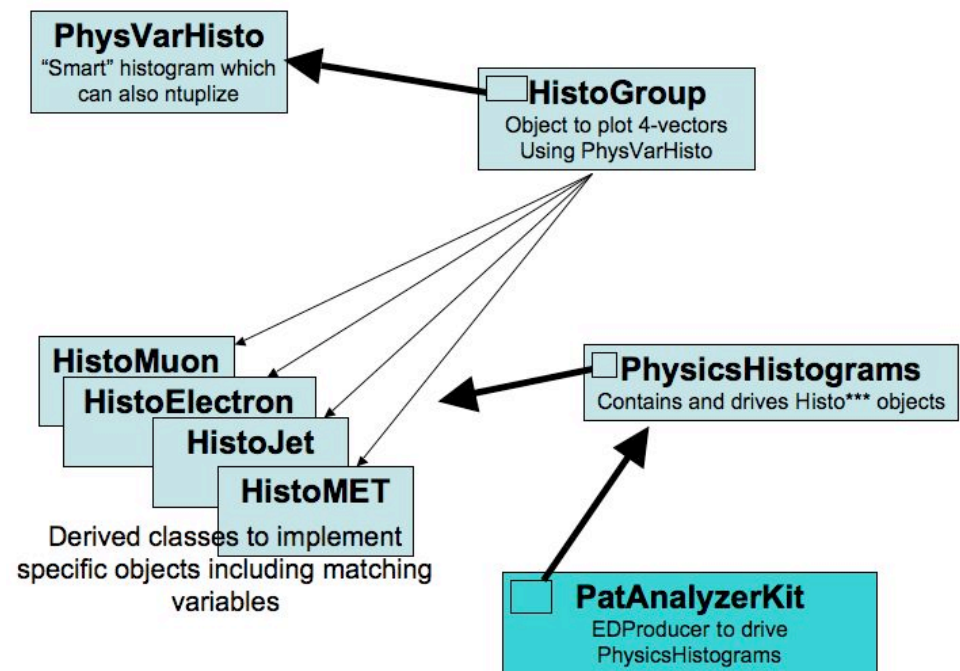
- automatically plot fourvector and object ID variables for all objects in the event

- sensible default axis limits

- also simple rootuples of these variables can be added

- driven by configuration files

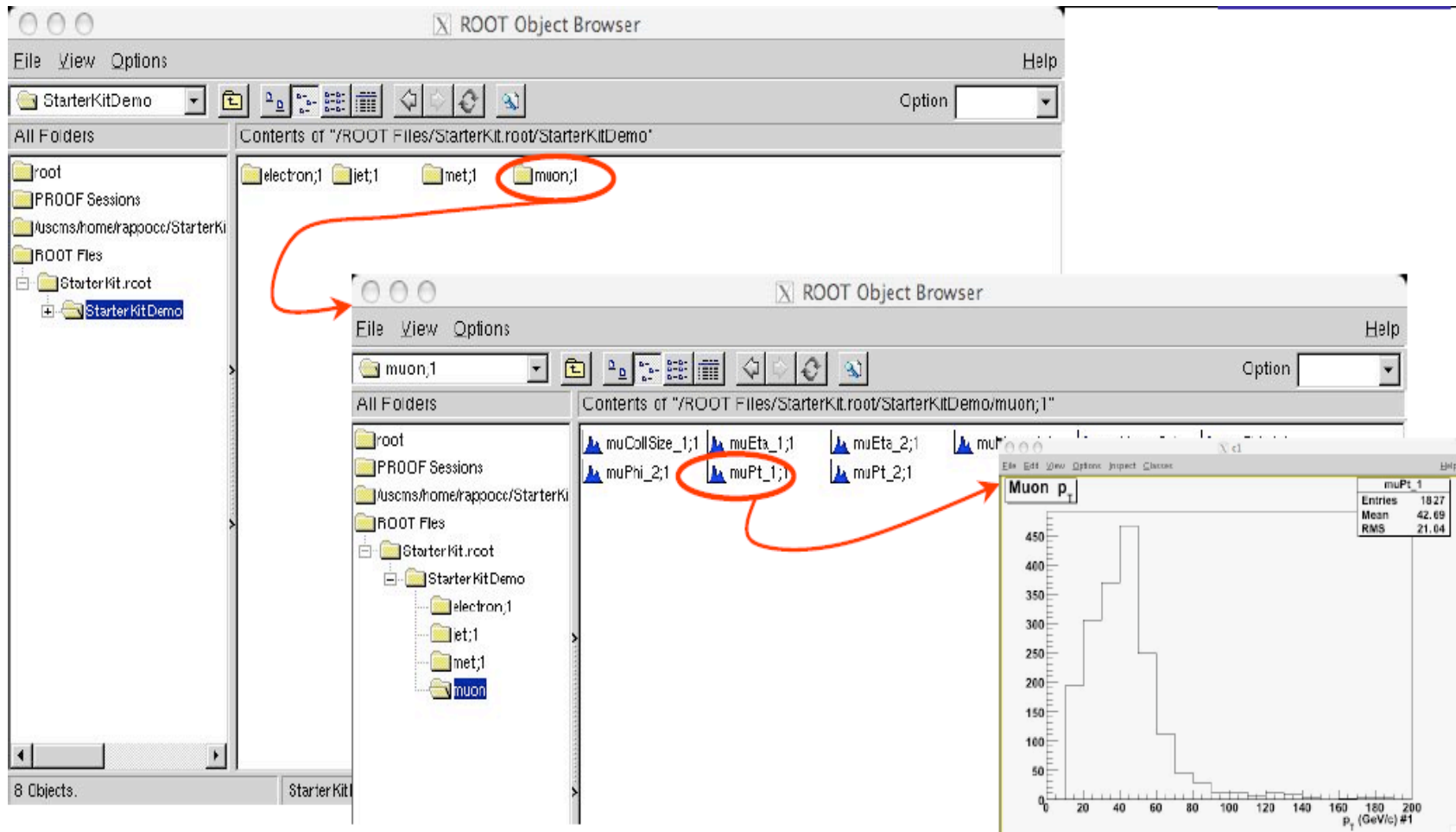
- ♦ check it out in the tutorial!





# The Starter Kit

Physics  
Analysis  
Toolkit





# The Physics Analysis Toolkit



## •Outline

- ◆ the Candidate model
- ◆ the Physics Analysis Toolkit
- ◆ the Starter Kit
- ◆ conclusions



# Conclusions

## •Conclusions

- ◆ the PAT is a vital project
  - started from and unifies the several CMS analysis frameworks
  - emphasis on both flexibility and user-friendliness is possible indeed
  - core PAT functionality is in good shape
- ◆ the PAT is ready for users
  - out-of-the-box examples available
  - documentation coming into place
  - embedded successfully in Layer-2 and Starter Kit
  - users/testers from all PAGs coming in
  - first analysis results being produced with PAT
- ◆ there is still a to-do list of course
  - we keep a prioritized list linked to the twiki
  - users are encouraged to contribute!
    - > not all tasks have names assigned
    - > add your own tools for existing/missing features!



# Documentation



## • Documentation and support

### ♦ documentation

- currently PAT's highest priority!
- with help from the CMS documentation people (Kati et al.)
- central wiki page in the CMS Software Guide:  
<https://twiki.cern.ch/twiki/bin/view/CMS/SWGuidePAT>  
and many links therein
- Starter Kit workbook to get you started  
<https://twiki.cern.ch/twiki/bin/view/CMS/WorkBookAnalysisStarterKit>
- PAT page in the CMS workbook: contains the PAT tutorial  
<https://twiki.cern.ch/twiki/bin/view/CMS/WorkBookPAT>
- see also the more general Physics Tools page:  
<https://twiki.cern.ch/twiki/bin/view/CMS/SWGuidePhysicsTools>

### ♦ support

- all questions should currently go to the PhysicsTools hypernews
- not so many developers now, but the more users use PAT, the more help you will get from users themselves



# Tutorials

Physics  
Analysis  
Toolkit

.There will now be a step-by-step tutorial session to put this into practice!

- ◆ Starter Kit tutorial

- <https://twiki.cern.ch/twiki/bin/view/CMS/WorkBookAnalysisStarterKit>

- ◆ PAT tutorial

- <https://twiki.cern.ch/twiki/bin/view/CMS/WorkBookPAT>

- ◆ Please stay for these, they will make a lot of these concepts clear and get you started with functional examples!



# Backup Slides





# Overview of the PAT

Physics  
Analysis  
Toolkit

## .Quote from the PAT mandate

The present situation causes substantial and unnecessary duplication of work and results in ever-increasing confusion of the users. In particular, analysis beginners (who constitute the vast majority of the CMS physicists at this point in time) do not know where to start from to get their input for their analysis, and as a result their possible involvement towards physics studies is delayed.

A related issue of concern is the analysis of the (first) data. In the absence of a unified PAT, analyses rely heavily on the availability of experts from the various groups, resulting in unnecessary delays between data taking and running of the analyses. These delays would slow down the understanding of potential problems with the detector, the reconstruction chain, the analysis programs and could even have dire effects on the competitiveness of CMS.





# The Physics Analysis Toolkit



## •Outline

- ◆ the CMS analysis environment
- ◆ overview of the Candidate model
- ◆ overview of the Physics Analysis Toolkit
  - PAT Layer-0
  - PAT Layer-1
  - PAT Layer-2
- ◆ the Starter Kit
- ◆ conclusions



# PAT Layer-0

## •Design considerations

- ♦ remember we want to reconcile **maximal flexibility vs. maximal ease-of-use**
- ♦ every action in the following steps is required to be fully configurable from config files
  - so also switched on/off
- ♦ every algorithm in the following steps needs to be factorized from the CMSSW framework, such that it can be re-used within FWLite
  - this means that e.g. no interface to the algorithms should involve the configuration language, and no database access is required



# PAT Layer-0

## •Object-dependent ID and cleaning

- ◆ object identification (ID) variables

- produce if not on AOD yet
- using POG recommendations as defaults
- interface multiple algorithms
- provide possibility for custom ID

- ◆ object flagging and cleaning

- technically: a cleaner (EDProducer) calls a FW-independent selector
  - this factorizes algorithmic code from the rest
- use the ID variables to choose “clean objects”
- technical implementation with flags written in the 'status' of the object
  - also used in the next step of event-wide ambiguity resolution
- possibility to keep “clean”, “non-clean” and all objects
  - e.g. isolated vs. non-isolated electrons



# PAT Layer-0

## • Cross-object ambiguity resolution

- ♦ happens logically in the same processing steps as the Layer-0 cleaners
- ♦ both “clean”, “not-clean” and “all” collections can be used to check overlaps
- ♦ in photon cleaning
  - electron removal “bySeed” (default) or “bySuperCluster”
- ♦ in tau cleaning
  - functionality there, but all switched off by default
- ♦ in jet cleaning
  - current default: only electrons (but in addition e.g. taus can be used)
- ♦ sequence of cleaners determines the order of ambiguity resolving
  - default order: muons, electrons, photons, taus, jets
- ♦ once cleaning disambiguation is done MET can be easily recomputed



# PAT Layer-0

## • Production of additional associated information

- ♦ at the end of the Layer-0 processing, additional analysis-level tasks can be performed to produce information not present yet in the RECO/AOD
- ♦ this should involve all remaining tasks depending on the framework
  - leave no obstacles to use FWLite on Layer-0 output
- ♦ some of these tasks don't involve an event interpretation (can possibly run before the actual Layer-0)
  - retrieval jet energy scale factors
  - parton flavour determination for jets
  - jet-track association and jet charge
  - additional b- and tau-tagging
  - ...
- ♦ some tasks depend on the event interpretation
  - MC matching, trigger primitive matching
  - external MET recalculation?
  - ...



# The Physics Analysis Toolkit



## •Outline

- ◆ the CMS analysis environment
- ◆ overview of the Candidate model
- ◆ overview of the Physics Analysis Toolkit
  - PAT Layer-0
  - PAT Layer-1
  - PAT Layer-2
- ◆ the Starter Kit
- ◆ conclusions



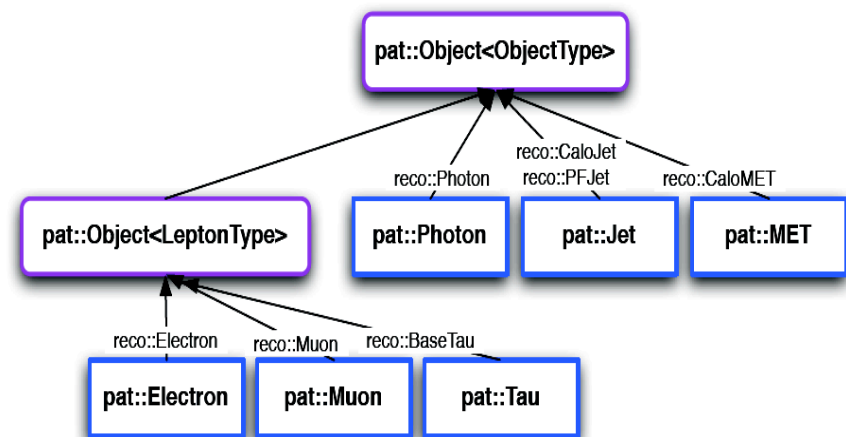
# PAT Layer-1

## •PAT objects: enriched reco dataformats

- ♦ PAT Layer-1 brings lots of related information together in “big” objects
  - easy one-entry user interface to previously associated information
  - 1-object approach without remaining associations makes it easy for users to play with the objects: event selection, scaling and smearing, vetoing, kinematic transformations, event hypothesis building, etc.

### ♦ PAT Layer-1 objects

- implemented:  
**Electron, Muon, Tau, Jet, Photon, MET**
- inherit from reco dataformats
- add additional members and methods for accessing the extra information
- only store what is asked for by the user



- ♦ will provide a uniform interface to Particle Flow objects



# PAT Layer-1

## •PAT object interfaces

- ◆ all PAT objects

- have methods to access their MC and trigger matches
- have methods to retrieve resolutions obtained from MC

- ◆ pat::Photon, pat::Electron, pat::Muon and pat::Tau

- contain methods to access isolation and ID variables

- ◆ pat::Jet

- provides access to b-tagging information
- methods for handling jet corrections
- access to jet flavour from MC
- interface to associated tracks and jet charge

- ◆ pat::MET

- MC match is in this case the generated MET





# The Physics Analysis Toolkit



## •Outline

- ◆ the CMS analysis environment
- ◆ overview of the Candidate model
- ◆ overview of the Physics Analysis Toolkit
  - PAT Layer-0
  - PAT Layer-1
  - PAT Layer-2
- ◆ the Starter Kit
- ◆ conclusions



## PAT Layer-2

### •Introducing event hypotheses

- ♦ up to now no assumptions have been made about the final state
  - in principle the final state hypothesis belongs more to the analysis
  - but tools and examples can be provided to users on how to easily deal with their specific final state
- ♦ User has a “physics process” in mind:

```
Z -> muon + muon
```

- ♦ Want to support structures that mimic this intuitive picture

```
muon1 : muon  
muon2 : muon  
Z = muon1 + muon2
```

- ♦ Can support multiple layers of hypothesis

```
H -> (Z -> muon + muon) + (Z -> electron + electron)
```

```
muon1 : muon  
muon2 : muon  
electron1 : electron  
electron2 : electron  
Z1 = muon1 + muon2  
Z2 = electron1 + electron2  
H = Z1 + Z2
```



## PAT Layer-2

### .Flat event hypothesis

- ◆ Provided by a simple list that associates strings to objects
- ◆ Access data members through those strings
  - Supports PERL-syntax  
regexpr

```
CandLooper looper<Muon> =  
hyp.loop("muon.");  
  
while( !looper() ) {  
    const Muon & mu = *looper;  
    fill( mu.pt() );  
    looper++;  
}
```

### .Hierarchical event hypothesis

- ◆ Uses a "named" Composite Candidate to store decay structures in a hierarchy
- ◆ Access data members through Candidate linked list utilities
- ◆ Added functionality: can access daughters by role string instead of index

```
const Candidate * Z1 = H.daughter("Z1");
```



## PAT Layer-2

### .“Hard coded” event hypotheses

- ♦ Top Quark Analysis Framework (TQAF)

- Has hard coded ttbar hypothesis

```
pat::Jet getHadb() const;  
pat::Jet getHadp() const;  
pat::Jet getHadq() const;  
pat::Jet getLepb() const;  
pat::Muon getMuon() const { return *muon_; };  
pat::Electron getElectron() const { return *electron_; };  
pat::MET getNeutrino() const { return *neutrino_; };
```

- Recently have interfaced TtSemiEvtSolution with NamedCompositeCandidates to provide generic access as a prototype

```
const reco::NamedCompositeCandidate & getRecoHyp() const { return recoHyp_; }
```

- ♦ Others are out there

